

SCANDIR

Vulnerable to TOCTOU issues

Sean Barnum, Cigital, Inc. [vita¹]

Copyright © 2007 Cigital, Inc.

2007-04-04

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 7954 bytes

Attack Category	<ul style="list-style-type: none">• Path spoofing or confusion problem	
Vulnerability Category	<ul style="list-style-type: none">• Indeterminate File/Path• TOCTOU - Time of Check, Time of Use	
Software Context	<ul style="list-style-type: none">• File Management	
Location	<ul style="list-style-type: none">• dirent.h	
Description	<p>The scandir() function scans the directory dir, calling filter() on each directory entry. Entries for which filter() returns non-zero are stored in strings allocated via malloc(), sorted using qsort() with the comparison function compar(), and collected in the array namelist, which is allocated via malloc(). If filter is NULL, all entries are selected.</p> <p>The alphasort() and versionsort() functions can be used as the comparison function compar(). The former sorts directory entries using strcoll(3), the latter using strverscmp(3) on the strings (*a)->d_name and (*b)->d_name.</p> <p>This function is in essence a TOCTOU security vulnerability. It can be used to return information about the directory structure of a system. If an attacker can select the value of dirname (due to the classic check/use scenario, then it is possible for him to determine what directories exist on a system.</p> <p>A call to scandir() should be flagged if the argument (the directory name) is used previously in a check-category call.</p>	
APIs	Function Name	Comments
	scandir	use
Method of Attack	<p>The key issue with respect to TOCTOU vulnerabilities is that programs make assumptions about atomicity of actions. It is assumed that checking the state or identity of a targeted resource followed by an action on that resource is all one action. In reality, there is a period of time between</p>	

1. <http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html> (Barnum, Sean)

	<p>the check and the use that allows either an attacker to intentionally or another interleaved process or thread to unintentionally change the state of the targeted resource and yield unexpected and undesired results.</p> <p>The scandir() call is a use-category call, which when preceded by a check-category call can be indicative of a TOCTOU vulnerability.</p>		
Exception Criteria			
Solutions	Solution Applicability	Solution Description	Solution Efficacy
	Applicable to most if not all uses of scandir();	Utilize a file descriptor version of check and use functions.	Generally effective.
	Applicable to most if not all uses of scandir();	The most basic advice for TOCTOU vulnerabilities is to not perform a check before the use. This does not resolve the underlying issue of the execution of a function on a resource whose state and identity cannot be assured, but it does help to limit the false sense of security given by the check.	Does not resolve the underlying vulnerability but limits the false sense of security given by the check.
	Applicable to most if not all uses of scandir();	Limit the interleaving of operations on files from multiple processes.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
	Applicable to most if not all uses of scandir();	Limit the spread of time (cycles) between the check and use of a resource.	Does not eliminate the underlying vulnerability but can help

			make it more difficult to exploit.
	Applicable to most if not all uses of scandir();	Recheck the resource after the use call to verify that the action was taken appropriately.	Effective in some cases.
Signature Details	<pre>int scandir(const char *dir, struct dirent ***namelist, int(*filter)(const struct dirent *), int(*compar)(const struct dirent **, const struct dirent **));</pre>		
Examples of Incorrect Code	<pre>/* print files in current directory in reverse order. No check first */ #include <dirent.h> main(){ struct dirent **namelist; int n; int check_status; struct stat statbuf; check_status=stat(fn, &statbuf); [...] n = scandir(".", &namelist, 0, alphasort); if (n < 0) perror("scandir"); else { while(n--) { printf("%s\n", namelist[n]- >d_name); free(namelist[n]); } free(namelist); } }</pre>		
Examples of Corrected Code	<pre>/* print files in current directory in reverse order. No check first */ #include <dirent.h> main(){ struct dirent **namelist; int n; n = scandir(".", &namelist, 0, alphasort); if (n < 0) perror("scandir"); else {</pre>		

	<pre>while(n--) { printf("%s\n", namelist[n]- >d_name); free(namelist[n]); } free(namelist); } }</pre>	
Source References	<ul style="list-style-type: none"> • Viega, John & McGraw, Gary. <i>Building Secure Software: How to Avoid Security Problems the Right Way</i>. Boston, MA: Addison-Wesley Professional, 2001, ISBN: 020172152X, ch 9 • UNIX man page for scandir() • http://maconlinux.net/linux-man-pages/en/scandir.3.html 	
Recommended Resource		
Discriminant Set	Operating Systems	<ul style="list-style-type: none"> • UNIX • Windows
	Language	<ul style="list-style-type: none"> • UNIX

Cigital, Inc. Copyright

Copyright © Cigital, Inc. 2005-2007. Cigital retains copyrights to this material.

Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works.

For information regarding external or commercial use of copyrighted materials owned by Cigital, including information about “Fair Use,” contact Cigital at copyright@cigital.com¹.

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. <mailto:copyright@cigital.com>